AD-A056 774    MISSISSIPPI STATE UNIV MISSISSIPPI STATE DEPT OF ELEC--ETC  F/G 9/3
COMPARISON OF REAL TIME AND FFT PROCESSING ALGORITHMS FOR ACOUS--ETC(U)
MAY 78  F M INGELS    DAAG29-77-G-0230

UNCLASSIFIED    MSSU-EIRS-EE-78-4    ARO-15462.1-R-EL    NL

1 OF 1
ADA
056774

END
DATE
FILMED
9-78
DDC

AD A056774

LEVEL II ARO 15462.1-R-EL

(12)

Approved for Public Release:
Distribution Unlimited

eirs

ENGINEERING & INDUSTRIAL RESEARCH STATION

ELECTRICAL ENGINEERING-MISSISSIPPI STATE UNIVERSITY



# COMPARISON OF REAL TIME AND FFT PROCESSING ALGORITHMS FOR ACOUSTIC DATA

by

FRANK M. INGELS



Final Report

MSSU-EIRS-EE-78-4

# COLLEGE OF ENGINEERING ADMINISTRATION

**HARRY C. SIMRALL, M.S.**
DEAN, COLLEGE OF ENGINEERING

**WILLIE L. MCDANIEL, JR., PH.D.**
ASSOCIATE DEAN

**WALTER R. CARNES, PH.D.**
ASSOCIATE DEAN

**LAWRENCE J. HILL, M.S.**
DIRECTOR, ENGINEERING EXTENSION

**CHARLES B. CLIETT, M.S.**
AEROPHYSICS & AEROSPACE ENGINEERING

**WILLIAM R. FOX, PH.D.**
AGRICULTURAL & BIOLOGICAL ENGINEERING

**JOHN L. WEEKS, JR., PH.D.**
CHEMICAL ENGINEERING

**ROBERT M. SCHOLTES, PH.D.**
CIVIL ENGINEERING

**B. J. BALL, PH.D.**
ELECTRICAL ENGINEERING

**W. H. EUBANKS, M.ED.**
ENGINEERING GRAPHICS

**FRANK E. COTTON, JR., PH.D.**
INDUSTRIAL ENGINEERING

**C. T. CARLEY, PH.D.**
MECHANICAL ENGINEERING

**JOHN I. PAULK, PH.D.**
NUCLEAR ENGINEERING

**ELDRED W. HOUGH, PH.D.**
PETROLEUM ENGINEERING

For additional copies or information
address correspondence to:

ENGINEERING AND INDUSTRIAL RESEARCH STATION
DRAWER DE
MISSISSIPPI STATE UNIVERSITY
MISSISSIPPI STATE, MISSISSIPPI 39762

TELEPHONE (601) 325-2266

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

| REPORT DOCUMENTATION PAGE | | READ INSTRUCTIONS BEFORE COMPLETING FORM |
|---|---|---|
| 1. REPORT NUMBER ~~MSSU-EIRS-EE-78-3~~ | 2. GOVT ACCESSION NO. | 3. RECIPIENT'S CATALOG NUMBER |
| 4. TITLE (and Subtitle) (6) COMPARISON OF REAL TIME AND FFT PROCESSING ALGORITHMS FOR ACOUSTIC DATA | | 5. TYPE OF REPORT & PERIOD COVERED (9) FINAL REPORT, 25 Sep[redacted]77 – 24 Jun[redacted]78 |
| | | 6. PERFORMING ORG. REPORT NUMBER MSU-EIRS-EE-78-4 |
| 7. AUTHOR(s) (10) Frank M. Ingels | | 8. CONTRACT OR GRANT NUMBER(s) (15) DAAG29-77-G-0230 |
| 9. PERFORMING ORGANIZATION NAME AND ADDRESS Mississippi State University, Electrical Engineering Department, P. O. Drawer EE, Mississippi State, MS 39762 | | 10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS P-15462-R-EL |
| 11. CONTROLLING OFFICE NAME AND ADDRESS U. S. Army Research Office, Post Office Box 12211, Research Triangle Park, NC 27709 | | 12. REPORT DATE May 30, 1978 |
| | | 13. NUMBER OF PAGES 48 |
| 14. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office) (11) 30 May 78 | | 15. SECURITY CLASS. (of this report) Unclassified |
| | | 15a. DECLASSIFICATION/DOWNGRADING SCHEDULE NA |

16. DISTRIBUTION STATEMENT (of this Report)

Approved for public release; distribution unlimited. (12) 58 p.

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

NA (18) ARO (19) 15462.1-R-EL

18. SUPPLEMENTARY NOTES

The findings in this report are not to be construed as an official Department of the Army position, unless so designated by other authorized documents.

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

Acoustic Signal Processing, FFT Algorithm Processor, Real Time Processor Algorithm, Computer Simulation of Processing Algorithms.

(14) MSSU-EIRS-EE-78-4

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

This report presents the numerical results and conclusions drawn from a computer simulation of two processing algorithms for acoustical data. The performance of the two algorithms, a real time processor (RTP) and a frequency domain processor (FFTP) is compared by the simulation and conclusions are drawn as to the relative merits of the two processing methods under various conditions of noise.

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE

Unclassified

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

410 772

COMPARISON OF REAL TIME AND FFT PROCESSING

ALGORITHMS FOR ACOUSTIC DATA

FINAL REPORT

Frank M. Ingels

June 10, 1978

U. S. ARMY RESEARCH OFFICE

Grant No. DAAG29-77-G-0230

Mississippi State University

## ABSTRACT

A computer model is developed for simulation of two acoustic signal processors: the real time processor (RTP) and the frequency domain processor (FFT).

Simulations are performed to assess the relative merits of the two processor techniques and comparisons of the possible performance of the two systems are presented.

## ACKNOWLEDGEMENTS

The author wishes to acknowledge the technical guidance provided by Captain H. Norckauer, Richard Currie and Steve Golden of the U.S. Army Missile Command and the patience of his wife for her endurance of these endeavors.

## TABLE OF CONTENTS

## LIST OF FIGURES

# LIST OF TABLES

# COMPARISON OF REAL TIME AND FFT PROCESSING ALGORITHMS FOR ACOUSTIC DATA

## 1.0 SUMMARY

Use of acoustic signals emanating from possible targets as a source for detection, location and tracking to these targets is of interest to the army RPV program.

Acoustic signals generated by the type of targets usually sought generally reside in the low audio frequency range. The ambient noise background is often considerable in this frequency range and it is necessary to provide a somewhat sophisticated algorithm for processing the acoustic signals.

Algorithms for processing these signals and determining the relative bearing of the targets have been developed through previous efforts at other agencies. These algorithms generally may be classified as one of two basic methods: Real Time processing and Frequency Spectrum Processing (generally a Fast Fourier Transform (FFT) application).

Although the FFT processing techniques have held the advantage until recently in so far as accuracy is concerned, recent state-of-the advances in software processing and integrated circuit chips (IC's) lead one to believe that a Real Time processing (RTP) algorithm for accuracy and at the same time update data faster than the FFT algorithm. Furthermore the RTP algorithm generally requires considerably less hardware than the FFT algorithms.

This report documents an investigative study of the two basic algorithms, RTP and FFT, and presents the results of a computer simulation study of their relative performance.

Under some circumstances the RTP is far superior; for instance in the situation where the signal frequency varies, such as an engine changing speeds. Both algorithms suffer inaccuracies for target angles off the end of the array (small angles) but this presents no real problems for a tracking application since the target will be kept in a broadside situation to the array (large angles) by the tracker.

In general it is concluded that a RTP algorithm is a very meritable approach and should be realized in a hardware prototype form to use in comparitive studies with an FFT algorithm hardware prototype.

This research work funded a graduate student working towards an MSEE degree and a paper describing the results of this research is being prepared for publication and presentation.

Much appreciation is expressed to Captain H. Norckauer, Richard Currie and Steve Golden of the Advanced Sensors Directorate (IR Branch) at Redstone Arsenal for their interest and suggestions.

## 2.0 INTRODUCTION

Acoustic signals emanating from targets such as tanks have been studied and their characteristics noted for use in detecting and tracking devices. One characteristic assumed for far field distances is the shape of the acoustic wavefront as it propagates outward. The wavefront is assumed to be planer, that is it is assumed to be an infinite plane wave (or a spherical wavefront of infinite radius). Thus the normal vector to the wavefront points to the source of the wavefront, which is assumed to be the target location. This normal vector describes an angle with respect to the axis of a two microphone array which may be determined by measuring the phase difference of the waveform as it crosses the two microphones.

The angle desired is $\theta$, illustrated in figure 1. This angle can be derived by several methods. If either real time waveforms or FFT algorithms are utilized and a periodic source assumed, then the angle is found by noting that

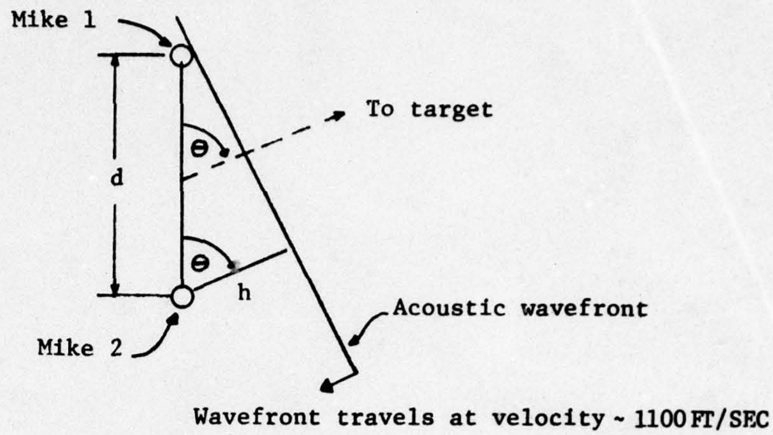
$$\cos\theta = \frac{h}{d}$$

where: $d$ = the array spacing

and $h = \tau V_{sd}$

with $V_{sd}$ the velocity of sound and $\tau$ the time difference in arrival of wavefront at microphones 1 and 2 and

$$\tau = \frac{\P(1/\text{freq.})}{(360°)} \quad .$$




O = Microphones

t = time to travel distance h at 1100 ft/sec = $\frac{h}{1100}$

$\phi$ Shift from Mike 1 to Mike 2 = $360° \times \frac{t}{T} = \frac{h \times 360°}{1100(T)} = \pi$

T = Period of the frequency components of acoustic wavefront (assumed to be ~ 100 Hz $\therefore$ T ~ .01 sec.)

$\theta$ = Target Angle = $\left( \cos^{-1} \left[ \frac{\phi \text{ Shift} \times 1100 \times T}{4 \times 360} \right] \right)$

Example: If $\phi$ Shift is calculated to be 100.28218° for 100 Hertz component with d = 4 feet for mike spacing, Then:

$$\theta = \left( \cos^{-1} \left[ \frac{100.28218 \times 1100 \times .01}{4 \times 360} \right] \right) = 40.000°$$

**Figure 1. Two Microphone Acoustic Array**

where ¶ is the phase difference of the waveforms at 1 and 2, and freq. is the frequency of the waveform. Thus

$$\P = \cos^{-1} \frac{\P(1/\text{freq.})V_{sd}}{d(360)}$$

inaccuracies arise from the following assumptions:

A. The acoustic wavefront is assumed to be a plane wave over a distance of a wavelength and to have a normal vector which passes through the centroid of the sound source and through the center of the microphone pair.

B. The acoustic wavefront is assumed to be a periodic single frequency wavefront. (Note the use of the frequency assumed in the equation for $\theta$ .)

C. The velocity of sound is assumed to be constant and known so that the distance, d, represents a certain percentage of the wavelength.

Items of Note:

(1) The spacing of the array is assumed to be such that no more than 180° phase shift will occur when the wavefront passes the array. Otherwise the array cannot differentiate which hemisphere the sound originates from with respect to a line drawn perpendicular to the line containing the two microphones. Thus the array spacing is determined in part by the highest frequency one expects to detect.

(2) A second array is necessary to determine which quadrant the sound originates from since a single two-microphone array does not determine the Quadrant.

It is feasible to determine the velocity of sound across the array by measuring the actual temperature as the phase is being measured, Item C. This point, incidentally, is a problem for both real time systems and for FFT systems. Measurement of the temperature at the array center provides sufficient correction for the calculation of the angle and it is noteworthy to mention that it is not necessary to determine the temperature along the path between the source and the detector since the velocity of the waveform is incidental to the calculations except during the time of travel from the first microphone in the array to the last microphone in the array. The assumption that the velocity of the wavefront is constant is valid since the array size is small and we are interested again in the velocity only as the wavefront crosses the array.

Both the real time and the FFT processors will suffer inaccuracies due to non-periodic components of the input, Item B. In the FFT processor, however, the noise and the varying spectral content of the signal are treated as periodic signals over each sampling period. As a result, these variations are averaged out to some extent and the spectral content of the target is enhanced with respect to the noise background.

The real time processor does not inherently average over any period and as a result the continuous output of phase difference information varies considerably due to noise and spectral variation of the signal. In a real time processor as designed by this author, however, the filter bandwidth may be considerably smaller than that of the FFT processor which does tend to reduce the noise effects considerably.

It is also beneficial to devise some sort of averaging scheme to use with the real time processor to improve the accuracy and to further reduce the effects of noise.

The following describes a software approach (now feasible with LSI and microprocessor technology) which provides an "averaging" effect for a real time process while allowing faster processing time than an FFT process.

The frequencies of interest are typically in the range of 100 Hertz. Using real time waveforms and a phase locked loop phase comparator the phase angle may be measured directly. It is proposed to sample the phase at certain intervals, say, .05-second intervals for ten samples. Let each phase sample be compared to the previous and if it agrees with another sample within, say, ten degrees, average the two and store. If it does not agree, store for further comparison as the possible target but with low priority. In this manner, the ten sample processor will discard large errors and average the small errors. After ten samples are used, an updating may be accomplished by simply keeping a running average and discarding the earliest data sample.

While it is true that this procedure will tend to average out valid target angle changes in .5 second intervals, the error at one mile would be 1.09° at the worst case and for shorter ranges should decrease since the bearing change would be kept relatively small by the tracking system.

A second and major problem arises when the target is an engine which is changing speed. This generates a signal which is changing in frequency. The FFT processor which uses long time averaging (from 0.5 second to 2.0 second intervals) will detect this type of signal as energy existing over a wide frequency spectrum and will exhibit large errors in the angular calculations of such a target.

On the other hand, the real time processor uses very short looking intervals and does not see appreciable frequency change. This results in fairly accurate angular calculations especially for angles around the null zone ($\theta = 90°$).

Results are presented which illustrate very nicely the ability of the real time processor to handle this situation and the inability of the FFT processor to deal with this problem.

One possible solution for the FFT processor to deal with this is to use a frequency tracking filter but this significantly adds to the cost and complexity of the system and has not been investigated to determine any benefits in performance.

Assumption that the normal vector to the acoustic wavefront passes through the centroid of the sound source, Item A, is not necessarily a valid assumption. Ripples in the wavefront, tilt due to propagation effects, multipath reception all combine to create errors in the normal vector pointing direction. These errors, however, are typically small, in the range of 5° or less, and are not troublesome during the tracking mode until one reaches a close or near field condition. In the near field it is questionable whether the sound centroid and the target itself will coincide.

Definitive studies of this problem have not been reported as yet.

The following sections of this report will describe computer programs that simulate the real time processor (RTP) and the fast Fourier transform processor (FFT) algorithms and presents results of simulation runs for various noise and signal situations.

# 3.0 MODELS USED FOR SIMULATION OF THE PROCESSORS

## 3.1 The RTP Model

The system diagram used to model the real time processor algorithm is illustrated in figure 2. The incoming acoustic wavefront is detected by the microphone array and is filtered by the 80 to 120 Hertz filter to eliminate as much background noise as possible. This filter design was based upon the acoustic signature for several tanks which were monitored by the Redstone facility during the summer of 1977. Experimental data taken during that period was correlated with acoustic signature data reported by other agencies and a center frequency of 100 hertz with a ± 20 Hertz passband was deemed to be fairly optimum, when one considers the possible frequencies of the source and the possible variation of those frequencies.

Figure 3 illustrates the simulated bandpass filter response which has a time response of

$$h(t) = 2(BW)\left[\mathrm{sinc}\left((BW)(t - t_o)\right)\right]\left[\cos\left(\omega_o(t - t_o)\right)\right]$$

where BW = bandwidth of the bandpass filter

and $$\mathrm{sinc}\ z = \frac{\sin \pi z}{\pi z}$$

The phase difference between the two microphones is measured by using a phase locked loop (PLL) circuit. As input noise and changer in the input. frequencies occur the phase information from the PLL will vary. These short term errors may be averaged out to some extent by using an algorithm for short term averaging. Such an

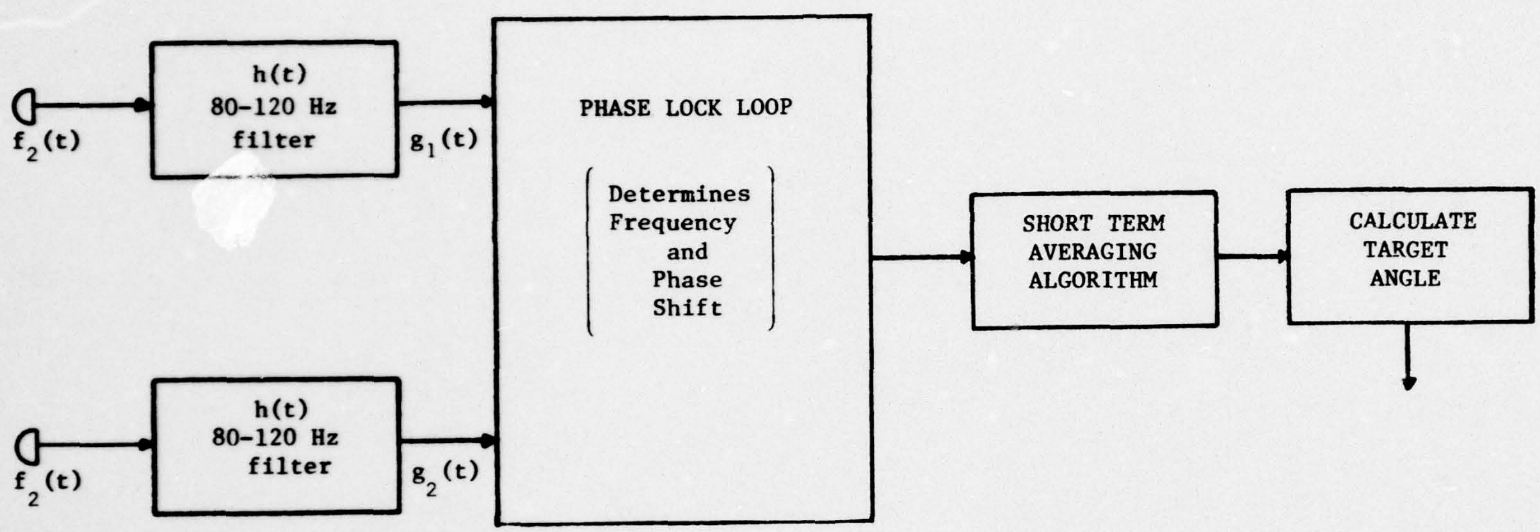



Figure 2. Real Time Processor System.

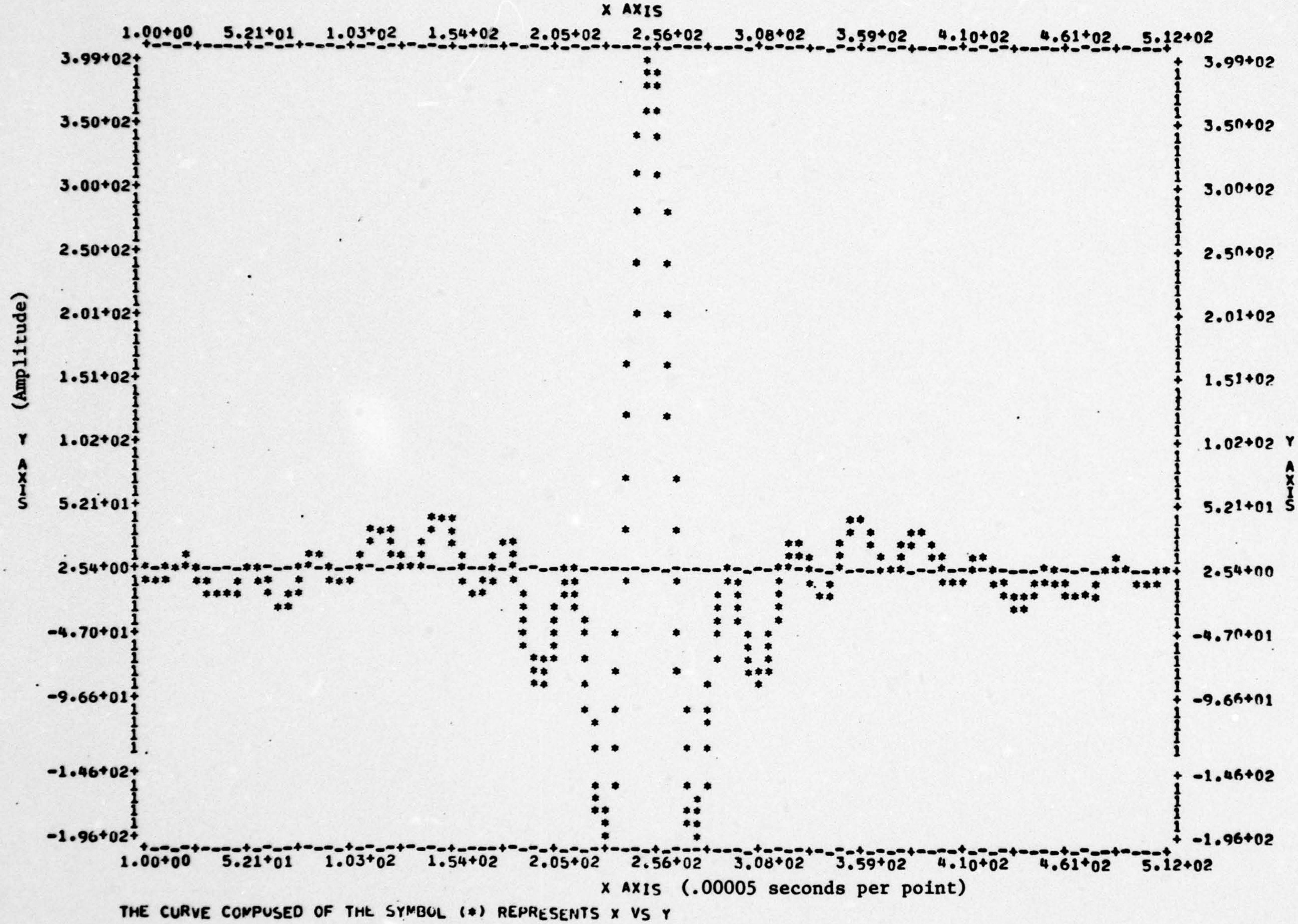



Figure 3. Bandpass Filter Simulation

algorithm has been proposed by the author and is detailed in the following discussion.

Signals in the frequency range of 100 Hertz are low enough in frequency to allow the phase lock loop to track minor $\pm$ 20 % variations in base frequency and phase in a time period negligible compared to the period of the 100 Hertz frequency (microseconds versus tens of milliseconds). A short term average may be computed by storing the frequency and phase difference of the incoming waveforms onice a period for four or five periods (approximately 40 - 50 ms.) and computing the average of these frequencies and phase differences.

This procedure will average out the short term variations of the input waveform to some extent. How well this procedure works may be judged by the results for the RTP which compare very favorably with the FFT processor. These results are described in sections 4.0 and 6.0.

Simulation of the data inputs from the microphones is accomplished by the use of a 1 volt peak cosine wave of 100 Hertz base frequency. Noise is added to this data input by use of a random number generator which is used to generate random amplitudes varying from +1 to -1. These numbers are generated on a uniform distribution and are used as additive noise. That is when the signal is generated and sampled for use in the program the noise, which has been generated is discrete steps, is added to the signal samples thereby producing a corrupted signal. Figures 4 through 7 illustrate the signal plus noise for various noise levels before and after filtering. The simulated signal plus noise may be illustrated mathematically by:

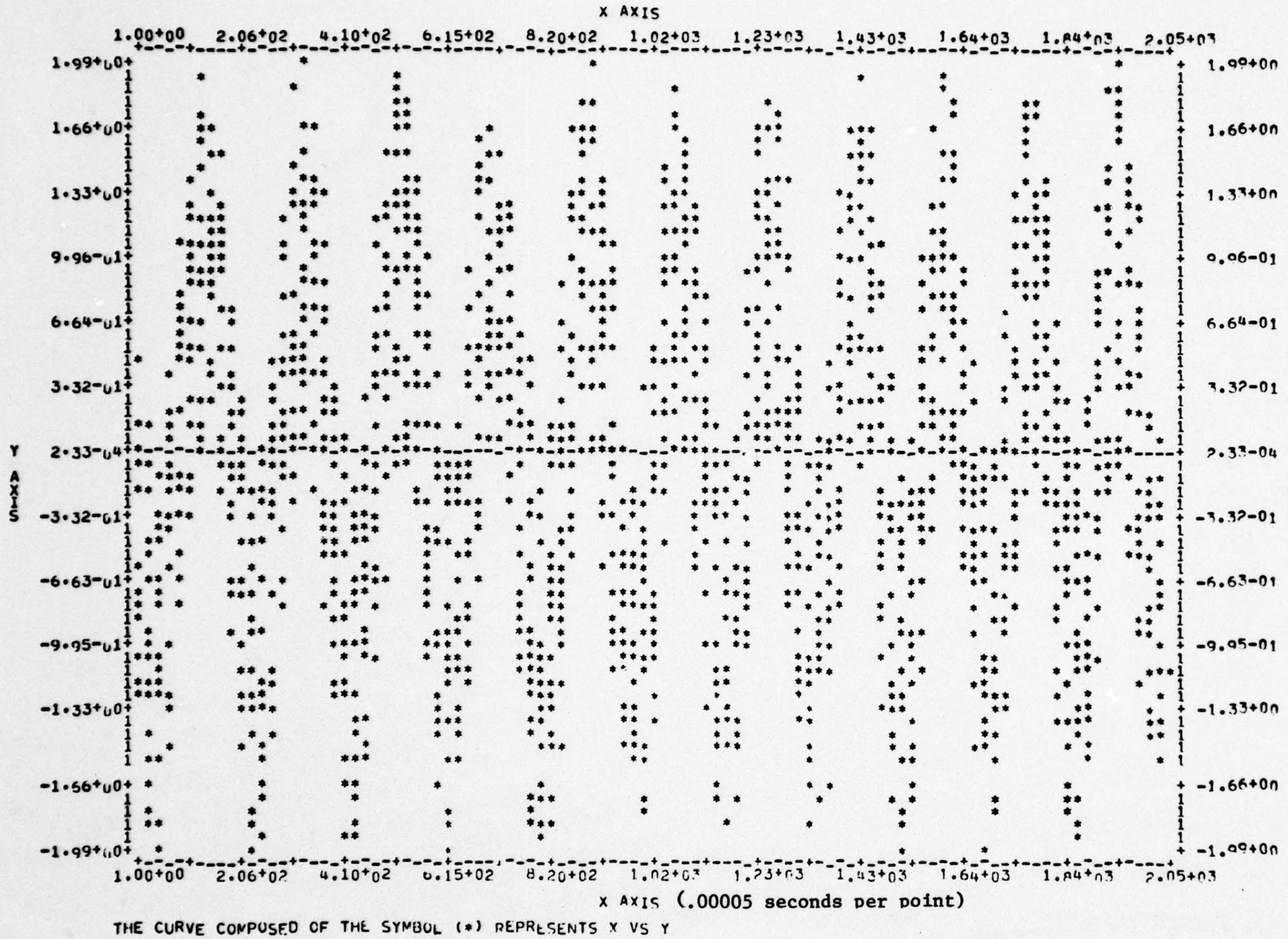



Figure 4. Signal Plus Noise Input $(S/N = 1.2225)_{rms}$ (100 Hertz signal)

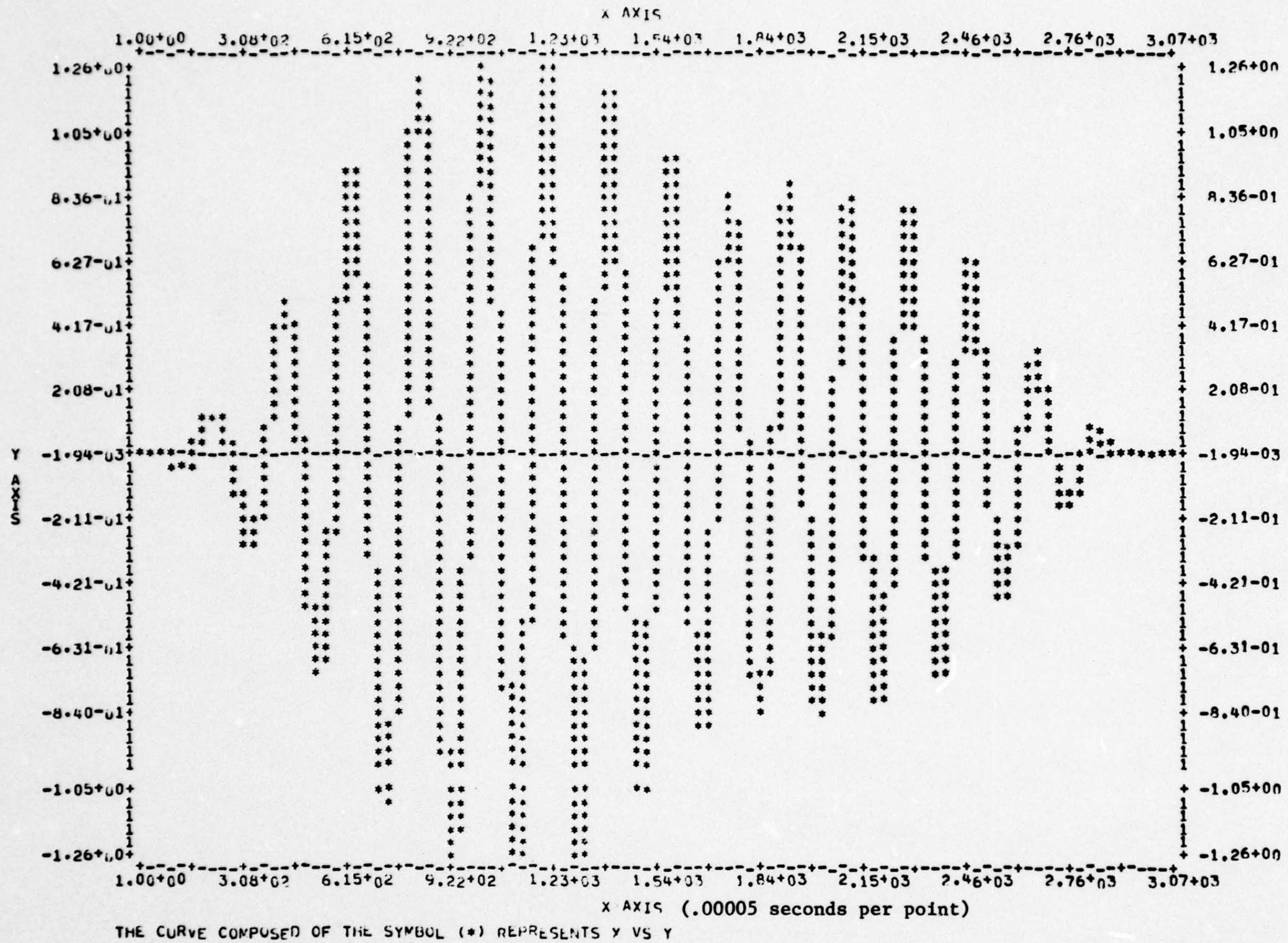



Figure 5. Signal Plus Noise Output $(S/N = 1.225)_{rms}$ (100 Hertz Signal)

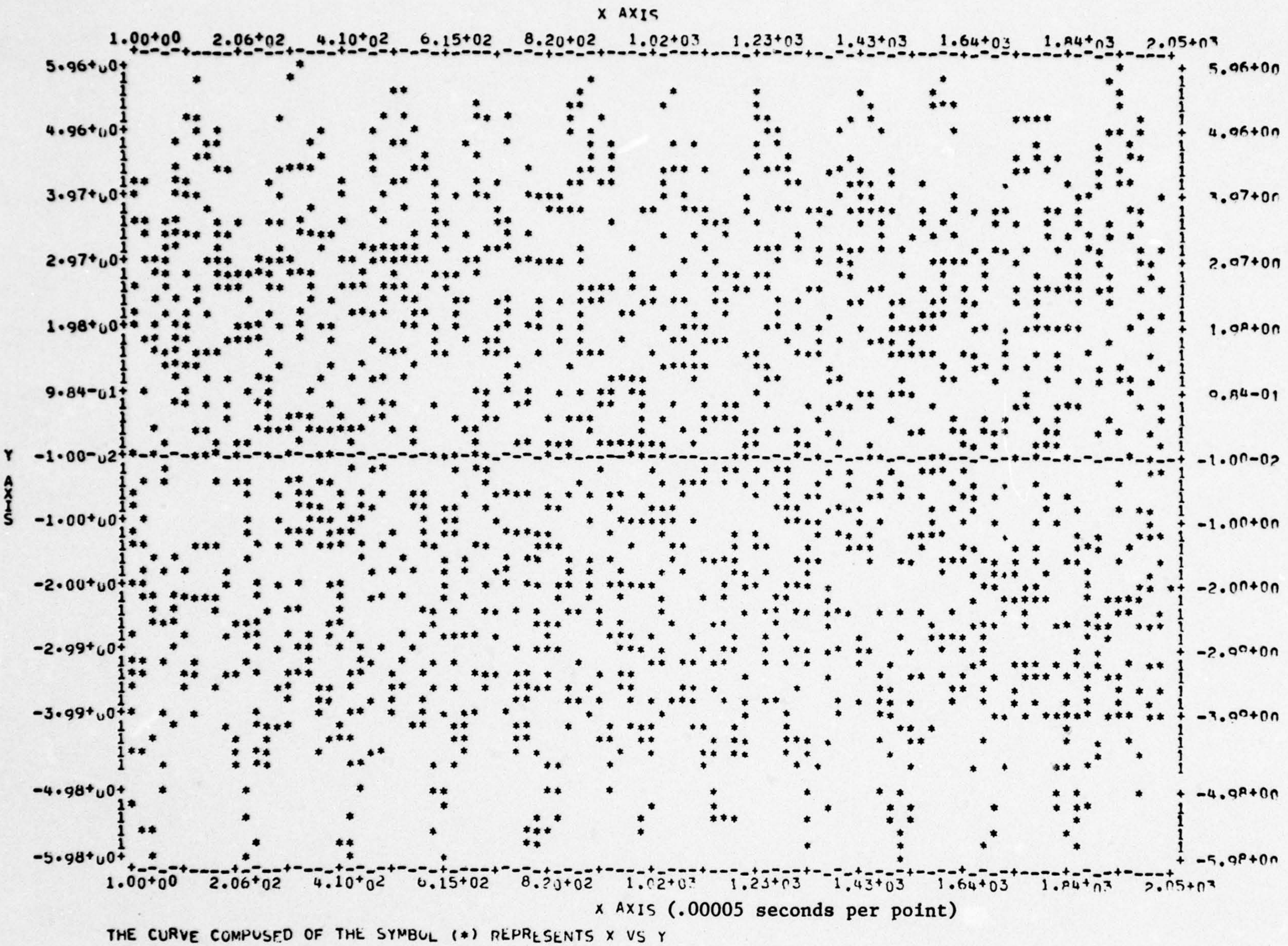



Figure 6. Signal Plus Noise Input $(S/N = 0.245)_{rms}$ (100 Hertz Signal)

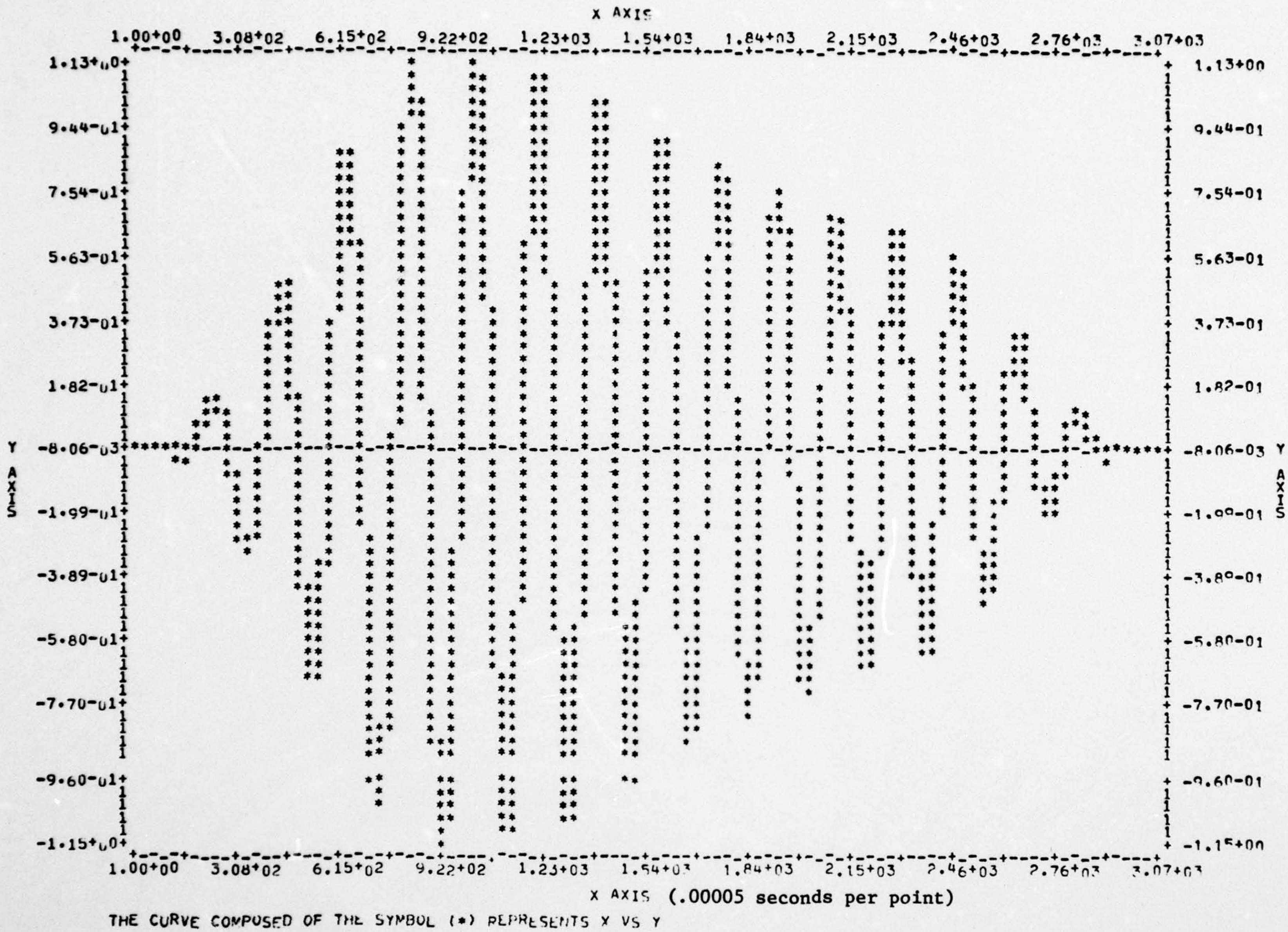



Figure 7. Signal Plus Noise $(S/N = 0.245)_{rms}$ (100 Hertz Signal)

$$f(t + \Delta\phi) = 1 \sin 2\pi\left((100 + \Delta f)t + \Delta\phi\right) + B \quad .$$

Signal frequency variation has been modeled by using a dither frequency term, $\Delta f$, added to the main signal frequency $f_o$. Thus $f_o + \Delta f$ is the signal frequency used and $\Delta f$ may be made to vary at any desired frequency range.

B is the random noise term and is uniformly distributed between levels b and a . The rms value of such a distribution is given by[2]

$$\sigma = \frac{(b - a)}{2\sqrt{3}}$$

when a and b are the upper and lower limits of the noise variations.

If the signal is a 1 volt peak sine wave then its rms value is .707 . Correspondingly if the noise is uniformly distributed between ± 5 volts then its rms value is 2.886 and the signal to noise ratio is

$$(S/N)_{rms} = \frac{.707}{2.886} = .2449757 \simeq .245 \quad .$$

For the simulation runs conducted a frequency variation of 10.24 Hertz (100 Hz to 110.24 Hz) at a 100 Hz/sec rate was used for the 0.1 second averaging RTP system.

The output g(t), figures 5 and 7 may be found by convolution of f(t) and h(t) through multiplication of their fourier transforms and inverse transforming the result to obtain g(t). One must be careful, however, to choose a proper sampling period and sampling interval so that the result for g(t) does in fact carry the information needed to represent g(t) in real time.

Since f(t) is composed of a periodic signal plus noise f(t) is not completely periodic. The FFT transform however forces one to pick a period over which to average and the function is assumed periodic with that period. As a result, the reproduction of g(t) by this method may not contain the non-periodic components of g(t).

A better method of simulating g(t) would be the use of the discrete convolution model. Since the bandpass filter is physically realizable and passive, its fourier transform may be determined analytically and hence its impulse response determined.

By computing g(t) through the discrete convolutional relationship

$$g(t) = \sum_{K=0}^{t} f(K)\, h(t - K)$$

a more faithful reproduction of g(t) will be obtained. The disadvantage of this method of simulation is the additional computation required. For instance, it has been shown (Stockham, T. G., "High Speed Convolution and Correlation," 1966 Joint Computer Conference) that for 28 samples the FFT method is faster and for 4096 camples the FFT method is 80 times faster. But with non-periodic inputs, the FFT method does not always produce accurate answers.

In programming the discrete convolution equation a method was devised that has good speed and a minimum of memory required. If one is not careful the memory requirements can be as high as M x N, where Mis is the number of samples of the filter function and N the number of samples of the signal plus noise function used. For 2048 samples of each the memory requirements would be greater than 4,194,304 words plus associated bookkeeping memory requirements. Obviously this is rather high (too high for most machines without resorting to time consuming tape storage). A method was devised that has essentially the same computational speed and requires only M + N (4096 words for 2048 samples) plus associated bookkeeping memory requirements.

The PPL phase comparator which will be modeled is a type II digital memory edge controlled network phase comparator. This type of phase comparator in this application will stay in lock over the full range of the $V_{co}$ (essentially from 5 Hz to 5 KHz). Phase comparison is made by cross referencing the two channels and using up-down counters to derive an 8 bit digital word which will indicate the phase difference between the two channels. The phase information is presented from a counter in digitized form in approximately 1.41 degree increments.

Simulation of the phase lock loop is accomplished by calculation of the frequency of the waveform coming from the filter h(t). This determination of the frequency is found by averaging the periods of the first six peaks in the output g(t) which is created by f(t) as the input. This average frequency is used in the calculation of the target angle.

The average phase shift is determined by comparing the shift of the main four peak values of the output $g(t + \Delta\phi)$, which is created by the signal input $f(t + \Delta\phi)$ as the input, relative to the main four peaks of g(t) created by f(t). This average phase shift is also used in calculating the target angle.

## 3.2 The FFT Model

The system diagram used to model the fast Fourier transform process is illustrated in figure 8. The filter bandwidth of 50-250 Hz was chosen so as to provide a frequency window in the frequency domain of the signal desired. This bandwidth could be restricted to a more narrow range; however, the system we are interested in investigating utilizes

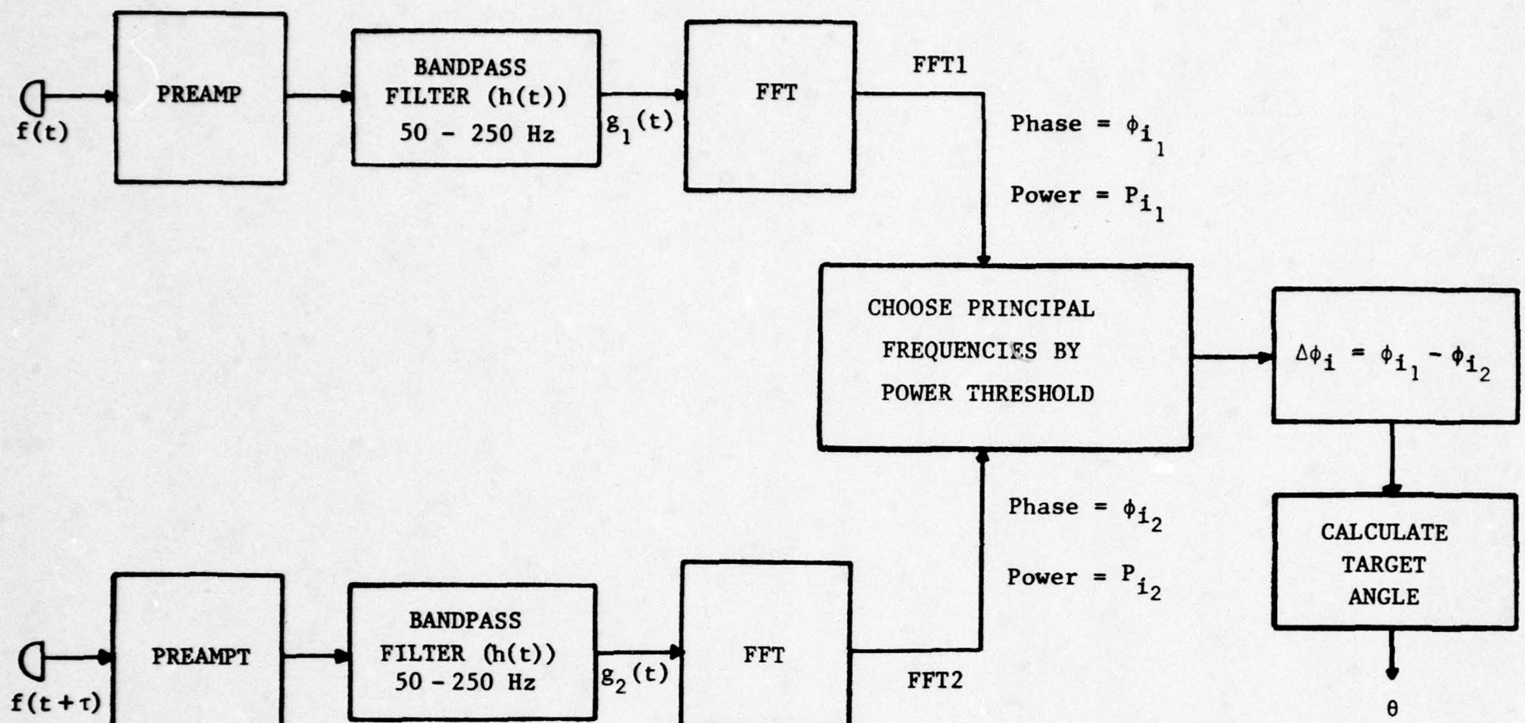



Figure 8. Basic FFT Processor

the 50-250 Hz bandwidth and hence the range was chosen for simulations.

The frequency domain plot will contain discrete frequency points (sometimes the nomencloture 'frequency bias' is used) and the spacing of these points (the width of the frequency bin) is inversely proportional to the period used for sampling. If a 2 second sample period is used the frequency points are 0.5 Hz apart.

The number of samples, N, taken per period, T, determines the sampling interval, $\tau = T/N$, which in turn relates to the width of the frequency spectrum calculated. This relation is NT and for the two-second sampling period of f(t) and 512 samples per period, the fast fourier transform will yield 512 frequency terms with each term .5 Hz wide. Thus, a frequency spectrum plot would cover from 0 to 256 Hz. (In a FFT half of the terms generated are complex conjugates of the others.)

The outputs of the fast fourier transform devices FFT1 and FFT2 are thus generated by first using discrete convolution to calculate the functions $g_1(t)$ and $g_2(t)$, and then a discrete FFT calculation is used to generate FFT1 and FFT2.

The outputs FF1 and FFT2 are complex numbers, one for each frequency bin. These complex numbers yield the amplitude of that frequency bin and its phase angle.

The amplitude is proportional to the energy content of the signal at this frequency. Thus if we have a strong signal energy at a specific frequency such as 100 Hertz and a very low noise background, we would expect to see a frequency plot that was very small except at 100 Hertz where we would expect to see a large amplitude component. Figure 9 illustrates a frequency plot for such a signal.

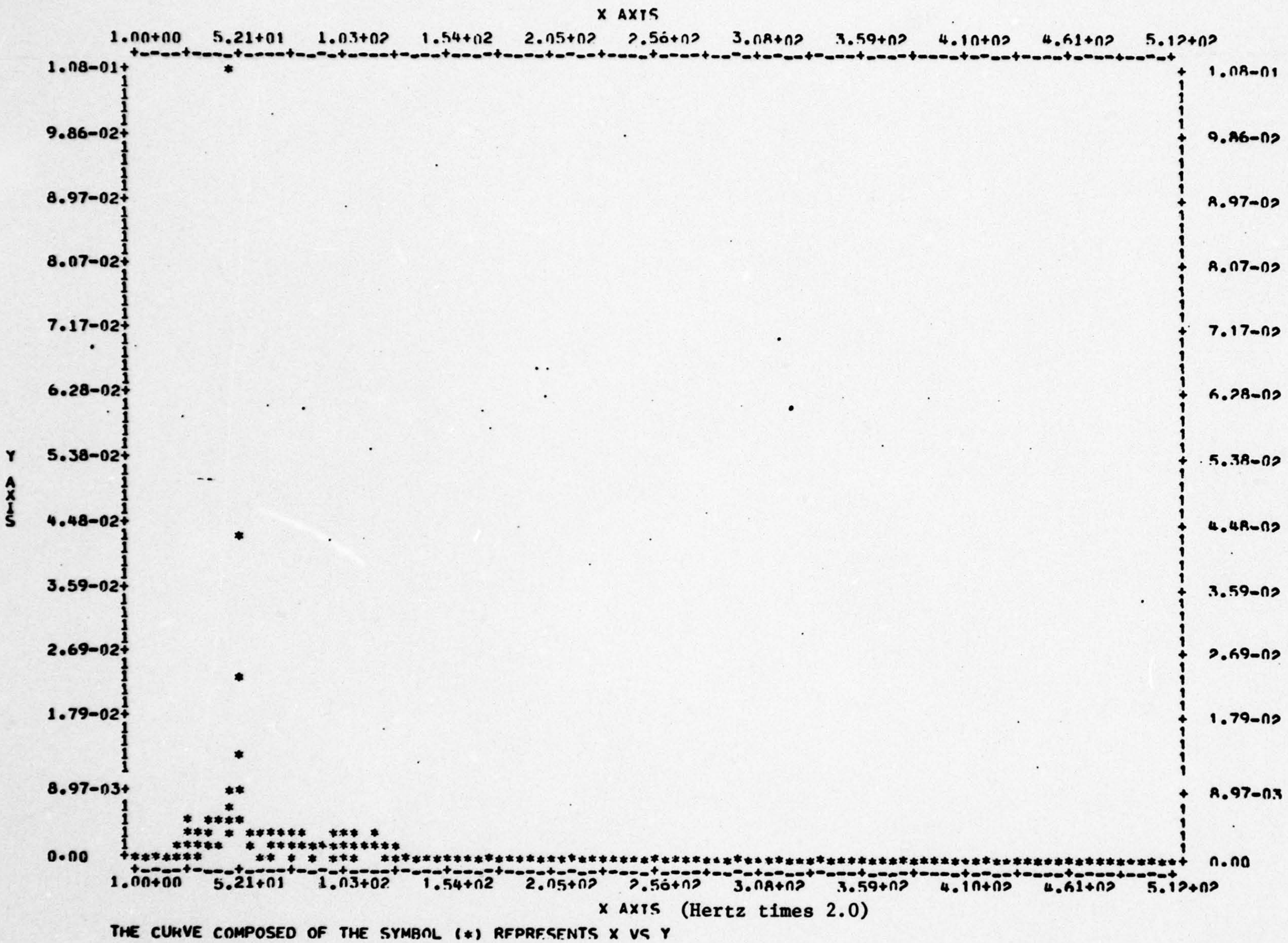



Figure 9. FFT Simulation Output for 100 Hertz Signal $(S/N = 1.225)_{rms}$.

Large amplitudes in specific frequency bins are selected and the phase angle of these terms are calculated. The phase shift of similar frequency components is then calculated by comparing the phase angle of a specific frequency bin for microphone 1 with the phase angle of the same specific frequency bin for microphone 2. The difference in the phase angle is proportional to the phase shift in that frequency term.

This technique is sufficient to simulate the system for these signal data inputs to be used and to compare the same basic information from the FFT and RTP systems. In general, however, one would refine the information from the FFT to allow target discrimination. This may be done as described.

The frequency bins are organized into groups by using a grouping algorithm which applies the principle that all frequency bins originating from the same direction in space will have essentially the same phase-difference slope. The phase-difference slope is defined as:

$$\text{Phase-difference slope} = \frac{\text{phase difference}}{\text{frequency}} = \frac{\Delta\phi_i}{f_i}$$

where i identifies the ith frequency bin.

The grouping is accomplished by identifying which slopes fall within a direction window in space.

The grouping algorithm step provides two important sources of data. It identifies the frequency lines coming from each target detected, which permits applying judgment as to the type of target present, as well as provides data to determine the best estimate of the target direction.

The best estimate of target direction is computed by averaging the phase-difference slopes making up each group.

$$\text{Average slope per group} = \frac{1}{n} \sum_{j=1}^{m} \frac{\Delta\phi_i}{f_i}$$

The target azimuth is then computed using the average slope per group and trigometric relationships.

$$\text{Azimuth} = \theta_A = \arcsin \frac{\sin \alpha}{\cos O_E}$$

where

$$\alpha = \arcsin \sum \frac{\frac{\Delta\phi_{ji}}{f_{ji}}}{n} \frac{k_2}{2\pi} \quad \text{for the microphone pair}$$

The subscript j refers to the target (group) number

The subscript i refers to the frequency bin number

$k_1, k_2$ = speed of sound in feet per second divided by the microphone separation in feet

$f_{ji}$ = frequency in Hertz for frequency bin i and target j

n = number of phase-difference slopes in the group.

Target discrimination is then provided both by the approach used and by performing additional checks.

## 4.0 RTP AND FFT SIMULATION RESULTS

The simulation of the RTP provides an insight into the possible performance of such a system.

Table 1 illustrates the performance of the RTP for various constraints within the simulation programs. For this table the random numbers used for the noise amplitude and for the algebraic sign were drawn from two separate random number generators. If different random number generators are used for the program while everything else is held constant, the individual results will vary but the average error should remain essentially constant. That is to say for one set of random numbers generating noise the angular results (for 40.0° actual target angle) may compute to be 41.1654°, while for a second set of random numbers the computed angle may be 38.710°. The results vary on an individual basis about the nominal 40.0° but the errors are within .3115% of each other.

The data in Table 1 illustrates several points. First of all, the fastest sample rate (.00005 seconds per sample) results in target angles that are generally closer to the actual target angle than the slower sample rates; this would be expected. Also true is the fact that for good results at small target angles, the faster sample rates are necessary. In fact, the quantization error due to slow sampling becomes prohibitive at .00025 seconds per sample time.

For the simulation areas comparing the FFT and RTP systems, the sample rate used was .00005 seconds per sample time.

In general, the results are indicative of a system algorithm that would be suitable for a tracker, especially for initial lockon and

TABLE 1. SIMULATION RESULTS FOR REAL TIME PROCESSOR

NOISE = UNIFORM ADDITIVE NOISE S/N = 1.0 (Peak to peak basis) or S/N = 1.225 (rms)
f(t) = 100 Hertz Signal

| AVG PERIOD (CALCULATED) | AVG PHASE SHIFT (CALCULATED) | ACTUAL PHASE SHIFT | ANGLE FOR TARGET (CALCULATED) | ACTUAL TARGET ANGLE |
|---|---|---|---|---|
| FOR: (.1024 seconds of output data -1024 samples of input -.00005 seconds sample time) | | | | |
| $.98875 \times 10^{-2}$ | 20.02529 | 22.73212 | 81.30063 | 80.00000 |
| $.98875 \times 10^{-2}$ | 64.17194 | 65.45450 | 61.00800 | 60.00000 |
| $.98875 \times 10^{-2}$ | 99.67131 | 100.2822 | 41.16540 | 40.00000 |
| $.98875 \times 10^{-2}$ | 123.3376 | 123.0143 | 21.32029 | 20.0000 |
| $.98875 \times 10^{-2}$ | 130.6195 | 130.9091 | 9.403374 | $.1563931 \times 10^{-1}$ |
| FOR: (.1024 seconds of output data - 512 samples of input - .0001 seconds sample time) | | | | |
| $.9925 \times 10^{-2}$ | 20.85642 | 22.73212 | 80.901920 | 80.00000 |
| $.9925 \times 10^{-2}$ | 61.66247 | 65.45450 | 62.12786 | 60.00000 |
| $.9925 \times 10^{-2}$ | 97.93451 | 100.2822 | 42.05518 | 40.00000 |
| $.9925 \times 10^{-2}$ | 121.5113 | 123.0143 | 22.89049 | 20.00000 |
| $.9925 \times 10^{-2}$ | 129.6725 | 130.9091 | 10.54077 | $.1563931 \times 10^{-1}$ |
| FOR: (.050 seconds of output data - 200 samples of input - .00025 seconds sample time) | | | | |
| $.9916667 \times 10^{-2}$ | 16.63886 | 22.73212 | 82.75909 | 80.00000 |
| $.9916667 \times 10^{-2}$ | 60.50420 | 65.45450 | 62.72039 | 60.00000 |
| $.991667 \times 10^{-2}$ | 99.83193 | 100.2822 | 40.86529 | 40.00000 |
| $.991667 \times 10^{-2}$ | 127.0588 | 123.0143 | 15.74056 | 20.00000 |
| $.9916667 \times 10^{-2}$ | 135.56485 | 130.9091 | | |

tracking. As the vehicle nears the target, an IR tracker with a narrow field of view could be used for the terminal homing. This would most likely be advisable since the IR tracker could provide a faster control loop in the guidance system.

From the data of Tables 1 and 2, it is evident that small variations in the phase shift and the inverse of frequency result in significant errors for small target angles. This is to be expected since the angle calculation is proportional to the inverse cosine of the phase shift and the inverse of the frequency. For small angles the cosine function has the fastest rate of change as opposed to the larger target angles which correspond to the slowest rate of change of the cosine function versus the argument of the function. Another way of saying this is to state the change of the argument of the inverse cosine function is slowest around the 90° area (the broad peak of the function) and quickest around the 0° area (the steepest slope), which may be interpreted as implying that small changes in the argument of the inverse cosine function yield proportionally larger changes in the resulting angle calculations for arguments corresponding to very small angles.

In any event, the results indicate (as we know to be true) that both types of algorithms make suitable null trackers by keeping the target broadside ($\theta = 90°$) to the axis of the array which operates in the most accurate region of the array.

TABLE 2. SIMULATION RESULTS FOR FFT PROCESSOR

NOISE = UNIFORM ADDITIVE NOISE  S/N = 1.0 (Peak to Peak Basis) or S/N = 1.225 (rms)
f(t) = 100 HERTZ SIGNAL  FILTER BANDWIDTH 50 - 250 Hertz

| AVG. FREQ. (CALCULATED) | AVG. PHASE SHIFT (CALCULATED) | ACTUAL PHASE SHIFT | ANGLE FOR TARGET (CALCULATED) | ACTUAL TARGET ANGLE |
|---|---|---|---|---|
| FOR: (2 seconds of output data - 1024 samples to FFT) G = 512 F = 3584 Dτ = .0005 | | | | |
| 100.5859 | 23.53366 | 22.73212 | 79.70457 | 80.0000 |
| 100.5859 | 65.02000 | 65.45450 | 60.40978 | 60.00002 |
| 100.5859 | 99.02980 | 100.2822 | 41.22991 | 40.00000 |
| 100.5859 | 121.4532 | 123.0143 | 22.72598 | 20.0001 |
| 100.5859 | 129.2968 | 130.9091 | 10.90868 | .015639 |
| FOR: (2 seconds of Output Data - 1024 Samples to FFT) G = 512 F = 1536 Dτ = .001 | | | | |
| 100.5859 | 21.76997 | 22.73212 | 80.48362 | 80.0000 |
| 100.5859 | 80.46195 | 100.2822 | 52.33385 | 40.0000 |
| FOR: (2 Seconds of Output Data - 1024 Samples to FFT) G = 1024 F = 1024 Dτ = .001 | | | | |
| 100.5859 | 18.08950 | 22.73212 | 82.10381 | 80.0000 |
| 100.5859 | 57.04849 | 123.0143 | 64.32611 | 20.00001 |

## 5.0 COMPARISON OF THE TWO PROCESSORS

The comparative results of the performance of the RTP and FFT processor algorithms are depicted in Table 3. The data in this table was made using a different random number generation method than that used to generate the data of Tables 1 and 2. However, the same random number generator was used for both the RTP and the FFT simulations of Table 3.

In this comparative run the two algorithms were used as they would be in an actual system. That is, the RTP system was presented with the signal plus noise data and an answer was computed in .15 seconds (actually the answer is computed using .08 seconds of the data emanating from the filter starting at $t = 0$), as opposed to the two seconds required by the FFT system.

The same signal data and the same noise data were used for both the RTP and the FFT calculations and the results indicate the relative performance under essentially the same input conditions.

Good agreement is obtained for both processor algorithms although the FFT processor is the more accurate of the two. In fact, for large noise levels, the FFT performs much better than the RTP, which was unable to arrive at a satisfactory answer for small target angles. However, if both systems are around the null point, both are useable for tracking. Another point to consider is the fact that as the vehicle approaches the target the signal-to-noise ratio will increase and likewise the accuracy of the processor algorithm increases in both cases.

Table 4 illustrates the results of a simulation in which the frequency of the incoming signal is varying. For this situation the

TABLE 3. SUMMARY OF SIMULATION RESULTS FOR RTP AND FFT PROCESSORS

NOISE = UNIFORM ADDITIVE NOISE S/N ON AN RMS BASIS AS INDICATED
f(t) = 100 Hertz Signal FILTER BANDWIDTH: RTP = 80 - 120 Hz FFT = 50 - 250 Hz
AVERAGING TIME: RTP = .15 seconds FFT = 2.0 seconds
SAMPLING INTERVAL: RTP = .00005 seconds/sample FFT = .0005 seconds/sample

| ACTUAL TARGET | RTP | | | | FFT | | | |
|---|---|---|---|---|---|---|---|---|
| ANGLE | NO NOISE | S/N = 1.285 | S/N = .612 | S/N = .245 | NO NOISE | S/N = 1.225 | S/N = .612 | S/N = 0.245 |
| 6.756° | 8.109° | - | - | - | 8.944° | 12.823° | 15.773° | 22.350° |
| 20.000° | 20.770° | 17.789° | 14.997° | - | 20.790° | 22.705° | 24.491° | 29.046° |
| 40.000° | 38.345° | 38.710° | 37.108° | 31.136° | 40.326° | 41.229° | 42.084° | 43.739° |
| 62.720° | 63.010° | 61.904° | 60.330° | 56.868° | 62.860° | 63.029° | 63.184° | 63.585° |
| 80.000° | 79.700° | 79.903° | 78.097° | 73.011° | 80.050° | 79.704° | 79.369° | 78.454° |

TABLE 4. SUMMARY OF SIMULATION RESULTS WITH FREQUENCY VARYING SIGNAL

FFT SIGNAL FREQUENCY VARIES FROM 100 Hz to 150 Hz at 27.90 Hz/sec RATE

2 Second Sample

S/N on an Rms Basis

| ACTUAL TARGET ANGLE | S/N = ∞ | S/N = 1.225 | S/N = .245 |
|---|---|---|---|
| 20.00001 | 28.05316 | 86.61458 | 48.41285 |
| 40.00000 | 44.06834 | 84.86287 | 46.37242 |
| 62.72039 | 64.7578 | 42.88194 | 70.14384 |
| 80.00000 | 80.78355 | 42.89643 | 67.31999 |

RTP SIGNAL FREQUENCY VARIES FROM 100 Hz to 110.24 Hz at 100 Hz/sec RATE

0.1 Second Sample

| ACTUAL TARGET ANGLE | S/N = ∞ | S/N = 1.225 | S/N = .245 |
|---|---|---|---|
| 20.00001 | 23.39165 | 23.39165 | 23.88287 |
| 40.00000 | 40.86529 | 40.56334 | 32.62816 |
| 62.72039 | 62.12786 | 61.68133 | 53.75424 |
| 80.00000 | 80.60277 | 77.69445 | 74.24273 |

FFT processor will suffer a degradation in performance that would render it unusable for a null tracker unless a considerable addition of software processing is added to the algorithm.

Figure 10 illustrates the spectrum which the FFT system "thinks" it sees when presented with a varying signal frequency. Since the FFT uses all data over the 2.0 second interval, it "thinks" that energy exists over the range of frequencies that the signal traverses during this time.

By comparison the RTP system holds up well since the signal frequency does not vary appreciably in the short processor time required.

For this simulation, the signal frequency was varied at a 100 Hertz per second rate for the RTP system and 27.90 Hertz per second rate for the FFT system.

Finally, in Table 5, are simulation results of the FFT system for a 0.5 second averaging time; which was proposed for a quicker updating system for close-in tracking.

Results indicate degradation of the system performance with short sample time for a constant signal frequency the results are useable for a high signal-to-noise ratio.

The results for a single frequency that varies, however, indicate that the system would not be useable since the output results are very erratic in accuracy (in fact, almost constant angle output regardless of target angle).

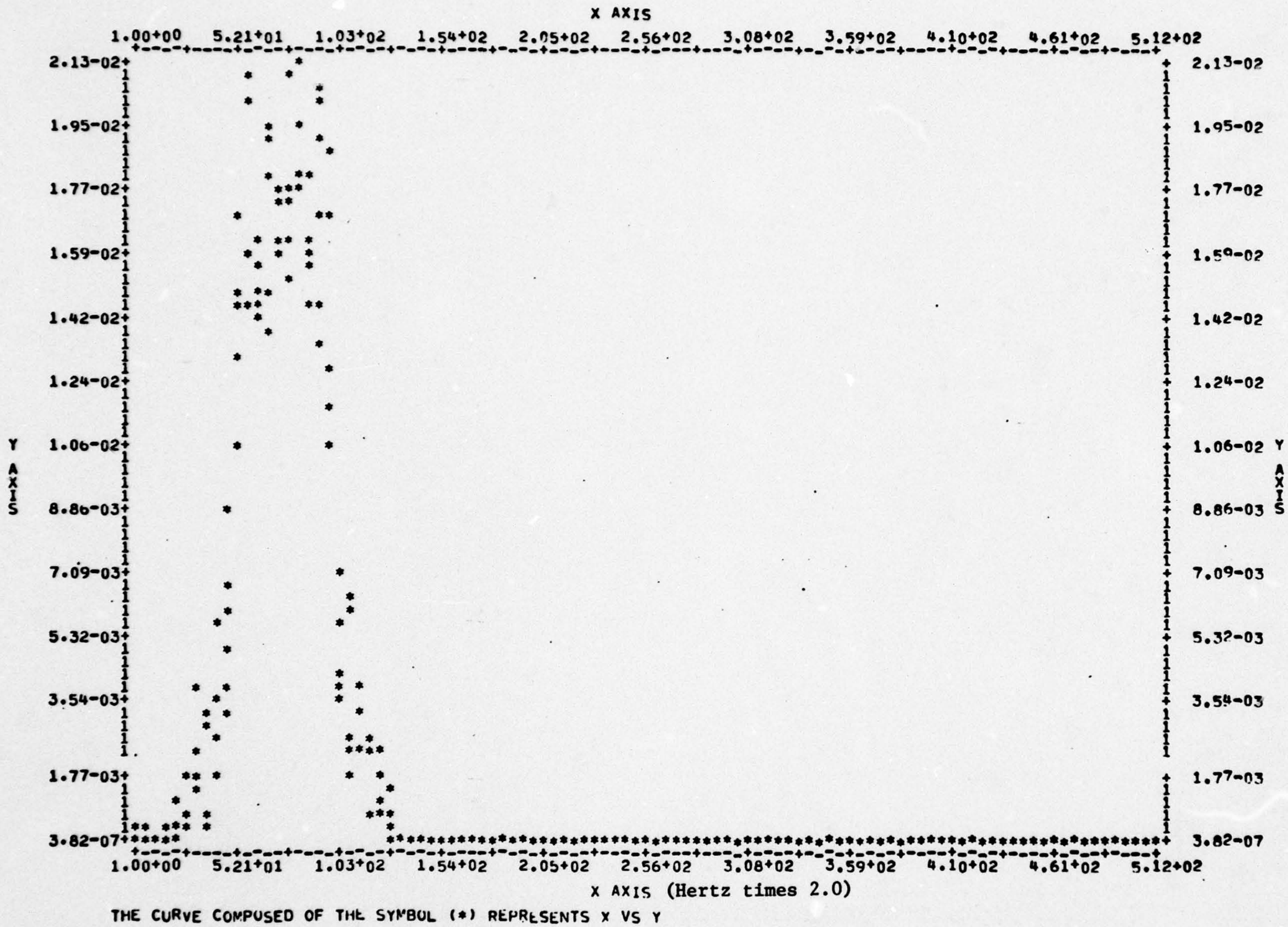



Figure 10. FFT Output for Varying Signal Frequency.

TABLE 5. SUMMARY OF SIMULATION RESULTS OF FFT FOR 0.5 SECOND AVERAGING

FFT WITH CONSTANT SIGNAL FREQUENCY OF 100 HERTZ

S/N on an RMS Basis

| ACTUAL TARGET ANGLE | S/N = ∞ | S/N = 1.225 | S/N = .245 |
|---|---|---|---|
| 6.756299 | 67.98864 | 57.05979 | 53.57781 |
| 40.00000 | 40.80123 | 66.23536 | 64.82656 |
| 62.72039 | 63.00268 | 76.48184 | 76.00817 |
| 80.00000 | 80.10936 | 85.04461 | 84.9896 |

FFT WITH SIGNAL FREQUENCY VARIATION FROM 100 to 150 HERTZ

| ACTUAL TARGET ANGLE | S/N = ∞ | S/N = 1.225 | S/N = .245 |
|---|---|---|---|
| 6.756299 | 78.51710 | 83.23922 | 85.57255 |
| 40.00000 | 71.39592 | 78.63190 | 89.18628 |
| 62.72039 | 82.89495 | 49.69523 | 85.93946 |
| 80.0000 | 81.88639 | 64.73020 | 76.22136 |

## 6.0 CONCLUSIONS

Computer programs have been written that simulate real time processing (RTP) and frequency domain (FFT) algorithms for processing acoustic data such as that emanating from tanks. Simulation tests of these systems have been accomplished and the results summarized in this report.

The simulation results indicate the following points:

1. The FFT system is the most accurate under conditions for which the signal frequency is constant.

2. The RTP system will provide a good tracking signal, especially for target angles of 30° or more.

3. The FFT system cannot tolerate a signal with varying frequency.

4. The RTP system is capable of processing the data approximately 10 times faster than the FFT system.

It seems feasible to design an acoustic tracking algorithm that utilizes a real-time processing algorithm. An RTP system holds several advantages over an FFT system, namely:

1. It is much less expensive due primarily to the simple hardware required with respect to an FFT system.

2. It is an order of magnitude faster in processing time that the FFT system which yields a faster guidance system control loop.

3. It has the ability to process and output useable data for signals with signal frequencies that are varying.

It is recommended that a real-time processing algorithm for processing acoustic signals such as those emanating from a tank be prototyped in breadboard form for evaluation in an actual environment under laboratory and controlled field test conditions.

It is further recommended that software be developed that is suitable for a microprocessor to average, categorize and perhaps provide a "smart" tracking algorithm for the RTP. Using this software development or tracker system using the RTP approach may be designed for a prototype tracker breadboard development.

## 7.0 REFERENCES


1. Stockham, T. G., "High Speed Convolution and Correlation," 1966 Joint Computer Conference.

2. Swartz, M., "Information Transmission, Modulation and Noise, McGraw Hill, 1959, Page 368.

APPENDICES

APPENDIX A

The following is a computer program written in fortram for simulation of the real time processor algorith, RTP. This program was run on a UNIVAC 1107 mod II system.

```
C      THIS PROGRAM IS THE SOLUTION TO THE REAL TIME PROCESSOR
C      H(T) IS THE IMPULSE REPONSE OF THE SYSTEM
C      X(T) IS THE INPUT SIGNAL TIME FUNCTION
C      THESE PARAMETERS ARE TO BE SET FOR EACH CASE
C      PTGAE = PERCENT TARGET ANGLE ERROR
C      SFO = SIGNAL FREQUENCY
C      SPO = SIGNAL PERIOD = 1/FREQUENCY
C      PHSF = ACTUAL PHASE SHIFT
C      ATARG = ACTUAL TARGET ANGLE
C      DT, F, G, IB, IFO
C      ENTER IB THE FILTER BANDWIDTH IN HERTZ
C      ENTER IFO THE FILTER CENTER FREQ IN HERTZ
C      ENTER DT
C      G MUST ALWAYS BE LESS THAN OR EQUAL TO F
C      ENTER G THE # OF DT INCREMENTS IN H(T)
C      ENTER F, THE # OF DT INCREMENTS IN X(T)
C      ENTER THE MIDPTS AMPTS OF DT FOR EACH DT IN H(T)
C      ENTER THE MIDPT AMPTS OF DT FOR EACH DT IN XT)
C      Y,V MUST BE DIMENSIONED TO (G+F)
       INTEGER G,F
       INTEGER Z
       INTEGER R
       G=1024
       F=2048
       DT=.00005
```

```
      DIMENSION PKYWM(400),PKYW(400),PKYM(400),PKY(400)
      DIMENSION XR(3584),X(3584),YW(4096)
      DIMENSION H(2048),V(3584)
C     VLS = VELOCITY OF SOUND IN FEET PER SECOND
      VLS = 1100.0
C     DARY = DISTANCE BETWEEN MICROPHONES ON ARRAY IN FEET
      DARY = 4.0
      SFO=100.0
      SPO = 1/SFO
C     THIS SETS NOISE TO 0 IF INS=0 OR GIVES NOISE IN INS=1
      INS=0.0
      PHSF=123.01430
C     THIS GENERATES THE DT MIDPT VALUES OF X(T) AND H(T)
      PI = 3.1415926
      JK = G+ F
      LLL = G+F
      IB=40
      IFO=100
      DO 110 JV=1,G
      V(JV)=JV
      VJ=(2*(JV-(G/2))+1)*DT/2.
      H(JV)=(2*IB*SIN(PI*IB*VJ)*COS(2*PI*IFO*VJ))/(PI*IB*VJ)
110   CONTINUE
      CALL PLOTS2(V,H,G,1)
      DO 117 IJV=1,2
      IG = G - 1.0
      JY = 1.0
      IJ = 1.0
      II = 1.0
      JJ=0.0
      IF(INS .GE. 1.0) GO TO 604
      AINS=100000.0
      DO 605 I=1,F
      XR(I)=0.0
605   CONTINUE
      GO TO 606
604   CONTINUE
      AINS=1.0/INS
      IF(IJV .EQ. 1.) GO TO 602
      XR(1)=385.
      GO TO 603
602   XR(1)=130.
603   CONTINUE
      CALL RANDU(XR,F)
606   CONTINUE
      DO 111 KV=1,F
      VK=(2*(KV-1)+1)*DT/2.
      PHI = (2*PI*PHSF)/360.
      IF(IJV .GT. 1.) GO TO 33
      GO TO 34
33    IF(XR(F-KV) .LT. 0.5) GO TO 600
      X(KV)=(COS(2*PI*(SFO+KV*.005)*VK))+(INS*XR(KV))
      GO TO 35
600   X(KV)=(COS(2*PI*(SFO+KV*.005)*VK))-(INS*XR(KV))
      GO TO 35
34    IF(XR(F-KV) .LT. 0.5) GO TO 601
      X(KV)=(COS((2*PI*(SFO+KV*.005)*VK)+PHI))+(INS*XR(KV))
      GO TO 35
601   X(KV)=(COS((2*PI*(SFO+KV*.005)*VK)+PHI))-(INS*XR(KV))
35    CONTINUE
111   CONTINUE
C     INSERT THE DO LOOP AND PLOTTING ROUTINE FOR   X  HERE
C     THIS REVERSES THE ORDER OF X(T)
      Z=F
      R=F/2.0
      DO 25 I=1,R
      T=X(I)
      X(I)=X(Z)
      X(Z)=T
      Z=Z-1
25    CONTINUE
      DO 10 K = 1,JK
      YW(K)=0.0
      IF(IJ .LE. G) GO TO 13
      GO TO 14
13    CONTINUE
      DO 20 I=1,IJ
      KI=(F+1.0)-I
      KY=(IJ+1.0)-I
      EUM=H(KY)*X(KI)*DT
```

```
      YW(K)=EUM+YW(K)
20    CONTINUE
      IJ=IJ+1.0
      GO TO 17
14    CONTINUE
      IF(JJ .GE. (F-G)) GO TO 16
      DO 15 JL=1,G
      KI=(F-JJ)-JL
      KY=(G+1)-JL
      EUM=H(KY)*X(KI)*DT
      YW(K)=EUM+YW(K)
15    CONTINUE
      JJ=JJ+1.0
      GO TO 17
16    CONTINUE
      II=II+1.0
      KI=(F+1)-II
      DO 21 I=1,IG
      KG=G+1.0-I
      JG=IG-I+1
      EUM=H(KG)*X(JG)*DT
      YW(K)=EUM+YW(K)
21    CONTINUE
      IG=IG-1
17    CONTINUE
10    CONTINUE
C     PUT PLOT ROUTINE FOR YW HERE
      DO 325 JIV=1,LLL
      V(JIV)=JIV
325   CONTINUE
      CALL PLOTS2(V,YW,LLL,1)
C     THIS FINDS THE PEAKS OF YW
      KJ=1.0
      PKYWM(1)=YW(1)
      PKYW(1)=1.0
      DO 313 J=2,LLL
      IF(YW(J) .GT. YW(J-1)) GO TO 313
      IF(J .EQ. 2) GO TO 313
      IF(YW(J-1) .GT. YW(J-2)) GO TO 314
      GO TO 313
314   PKYWM(KJ)=YW(J-1)
      PKYW(KJ)=J-1
      KJ=KJ+1
      KKJ=KJ
313   CONTINUE
C     THIS FINDS THE MAIN PEAK OF YW
      X=PKYWM(1)
      MPKYW=1
      DO 28 J=2,KKJ
      Y=PKYWM(J)
      IF(Y .GE. X) GO TO 29
      GO TO 28
29    X=Y
      MPKYW=J
28    CONTINUE
      IF(IJV .GT. 1.0) GO TO 126
      DO 421 J=1,KKJ
      PKY(J)=PKYW(J)
      PKYM(J)=PKYWM(J)
421   CONTINUE
      MPKY=MPKYW
      JJK=KKJ
126   CONTINUE
C     INSERT DO LOOP AND WRITE ROUTINES FOR Y AND YW HERE
C     Y IS THE PHASE SHIFTED SIGNAL AND IS COMPUTED FIRST
117   CONTINUE
      WRITE(6,319)
319   FORMAT(1H1,'PEAK VALUES OF YW EQUAL TO F(T)')
      WRITE(6,320) (PKYW(J), PKYWM(J),J=1,KKJ)
320   FORMAT(4(G12.7,2X,G12.7))
      WRITE(6,321)
321   FORMAT(1H0,'PEAK VALUES OF Y EQUAL TO F(T+PHI)')
      WRITE(6,322) (PKY(J),PKYM(J),J=1,JJK)
322   FORMAT(4(G12.7,2X,G12.7))
      SUM=0.0
C     THIS FINDS AVERAGE PERIOD USING 2 PEAKS EACH SIDE OF  MAIN PEAK
      DO 317 J=1,4
      KQ=MPKYW-2+J
      JQ=MPKYW-3+J
      DIFF = PKYW(KQ)-PKYW(JQ)
```

```
      SUM=SUM+DIFF
317   CONTINUE
      AVPRD=(SUM/4.0)*DT
      FREQ = 1/AVPRD
      SMPAS=0.0
C     THIS FINDS AVERAGE PHASE SHIFT USING 2 PEAKS EACH SIDE OF MAIN
      DO 318 J=1,4
      KQ=MPKY-2+J
      JQ=MPKYW-2+J
      PASDF=((PKY(KQ)-PKYW(JQ))*DT*360.0)/AVPRD
      SMPAS=SMPAS + PASDF
318   CONTINUE
C     THIS CHECKS THE AVPAS FOR 360 ROTATION
      IF(SMPAS) 60,61,61
60    AVPAS=ABS(SMPAS)/4.0
      GO TO 62
61    IF(SMPAS .LE. 1440.0) GO TO 620
      SMPAS=SMPAS-1440.0
      GO TO 61
620   AVPAS=360.0-(SMPAS/4.0)
62    CONTINUE
      IF(AVPAS .LE. 360.0) GO TO 63
      AVPAS = AVPAS - 360.0
      GO TO 62
63    CONTINUE
      WRITE(6,44)
44    FORMAT(1H0,'MPKYW',12X,'MPKY',13X,'AVPRD',12X,'AVPAS')
      WRITE(6,45) (MPKYW,MPKY,AVPRD,AVPAS)
45    FORMAT(G12.7,5X,G12.7,5X,G12.7,5X,G12.7)
C     THIS PROTECTS AGAINST INVALID ACOS ARGUMENTS
      ARGMT=(AVPAS*VLS*AVPRD)/(DARY*360.0)
      BV=ABS(ARGMT)
      IF(BV .LT. 1.0) GO TO 40
      ANGLE=999.99
      ATARG=999.99
      PTGAE=999.99
      GO TO 42
40    ANGLE=(ACOS(BV))*(360.0/(2*PI))
      ATAMT=(PHSF*VLS*SPO)/(DARY*360.0)
      BB=ABS(ATAMT)
      IF(BB .LT. 1.0) GO TO 43
      ATARG=999.99
      PTAGE=999.99
      GO TO 42
43    ATARG=(ACOS(ATAMT))*(360.0/(2*PI))
      PTGAE=((ATARG-ANGLE)/ATARG)*100.0
42    CONTINUE
      WRITE(6,326)
326   FORMAT(1H0,'RTP WITH S/N = 1/INS')
      WRITE(6,327) AINS
327   FORMAT(G13.6)
      WRITE(6,323)
323   FORMAT(1H0,' FREQUENCY',5X,'AVG PHASE SHIFT',5X,'ACTUAL PHASE SHIF
     /T',5X,'ANGLE TO TARGET',5X,'ACTUAL TARGET ANGLE',5X,'% TARGET ANGL
     /E ERROR',/)
      WRITE(6,324) (FREQ,AVPAS,PHSF,ANGLE,ATARG,PTGAE)
324   FORMAT(E13.6,4X,G12.7,6X,G12.7,11X,G12.7,8X,G12.7,13X,G12.7)
      STOP
      END
```

APPENDIX B

The following is a computer program written in fortram for simulation of the frequency domain processor algorith, FFT. This program was run on a UNIVAC 1107 mod II system.

```
C*    THE NAME PKYWM APPEARS IN A DIMENSION OR TYPE STATEMENT BUT IS NEVER REFEREN
C*    THE NAME PKYW APPEARS IN A DIMENSION OR TYPE STATEMENT BUT IS NEVER REFERENC
C         THIS PROGRAM IS THE SOLUTION TO THE FFT PROCESSOR
C         H(T) IS THE IMPULSE REPONSE OF THE SYSTEM
C         X(T) IS THE INPUT SIGNAL TIME FUNCTION
C         THESE PARAMETERS ARE TO BE SET FOR EACH CASE
C         PIGAE = PERCENT TARGET ANGLE ERROR
C         SFO = SIGNAL FREQUENCY
C         SPO = SIGNAL PERIOD = 1/FREQUENCY
C         PHSF = ACTUAL PHASE SHIFT
C         ATARG = ACTUAL TARGET ANGLE
C         DT, F, G, IB, IFO
C         ENTER IB THE FILTER BANDWIDTH IN HERTZ
C         ENTER IFO THE FILTER CENTER FREQ IN HERTZ
```

```
C       ENTER DT
C       G MUST ALWAYS BE LESS THAN OR EQUAL TO F
C       ENTER G THE # OF DT INCREMENTS IN H(T)
C       ENTER F, THE # OF DT INCREMENTS IN X(T)
C       ENTER THE MIDPTS AMPTS OF DT FOR EACH DT IN H(T)
C       ENTER THE MIDPT AMPTS OF DT FOR EACH DT IN XT)
C       Y,V MUST BE DIMENSIONED TO (G+F)
        INTEGER G,F
        INTEGER Z
        INTEGER R
C       IE = THE POWER OF 2 FOR THE FFT ROUTINE NUFFT(IE)
        IE=10
C       XR = THE NUMBER OF NOISE TERMS TL BE ADDED TO THE SIGNAL X
C       IH = NUMBER OF POINTS FOR FFT TO OPERATE ON
        IH=1024
        G=512
        F=3584
        DT=.000125
        DIMENSION PKYWM(400),PKYW(400)
        DIMENSION FPKFT(400),FPKNB(400)
        DIMENSION XR(3584),YW(4096),X(3584),V(3584)
        DIMENSION H(1024),Y(1024)
        DIMENSION AYW(1024),FIY(1024),PKFFT(400),PKFNB(400)
        DIMENSION FAB(1024),FI(1024)
        DIMENSION FYW(1024)
        COMMON/HOOK/XREAL(1024),XIMAG(1024)
        EQUIVALENCE (XREAL(1),AYW(1)),(XIMAG(1),FI(1))
C       VLS = VELOCITY OF SOUND IN FEET PER SECOND
        VLS = 1100.0
C       DARY = DISTANCE BETWEEN MICROPHONES ON ARRAY IN FEET
        DARY = 4.0
        SFO=100.0
        SPO = 1/SFO
C       THIS SETS NOISE TO  ZERO IF INS = 0 OR GIVES NOISE IF INS = 1.0
        INS=1.0
        PHSF=130.000
C       THIS GENERATES THE DT MIDPT VALUES OF X(T) AND H(T)
        PI = 3.1415926
        JK = G+ F
        LLL = G+F
        IFO=150
        IB=200
        DO 110 JV=1,G
        V(JV)=JV
        VJ=(2*(JV-(G/2))+1)*DT/2
        H(JV)=(2*IB*SIN(PI*IB*VJ)*COS(2*PI*IFO*VJ))/(PI*IB*VJ)
110     CONTINUE
        CALL PLOTS2(V,H,G,1)
        DO 117 IJV=1,2
        DO 53 J=1,IH
        FI(J)=0.0
53      CONTINUE
        IG = G - 1.0
        JY = 1.0
        II = 1.0
        IJ = 1.0
        JJ=0.0
        IF(INS .GE. 1.0) GO TO 604
        AINS=100000.0
        DO 605 I=1,F
        XR(I)=0.0
605     CONTINUE
        GO TO 606
604     CONTINUE
        AINS=1.0/INS
        IF(IJV .EQ. 1.) GO TO 602
        XR(1)=385.
        GO TO 603
602     XR(1)=130.
603     CONTINUE
        CALL RANDU(XR,F)
606     CONTINUE
        DO 111 KV=1,F
        VK=(2*(KV-1)+1)*DT/2.
        PHI = (2*PI*PHSF)/360.
        IF(IJV .GT. 1.) GO TO 33
        GO TO 34
33      IF(XR(F-KV) .LT. 0.5) GO TO 600
        X(KV)=(COS(2*PI*(SFO+KV*.01395)*VK))+(INS*XR(KV))
        GO TO 35
```

```
600     X(KV)=(COS(2*PI*(SFO+KV*.01395)*VK))-(INS*XR(KV))
        GO TO 35
34      IF(XR(F-KV) .LT. 0.5)  GO TO 601
        X(KV)=(COS((2*PI*(SFO+KV*.01395)*VK)+PHI))+(INS*XR(KV))
        GO TO 35
601     X(KV)=(COS((2*PI*(SFO+KV*.01395)*VK)+PHI))-(INS*XR(KV))
35      CONTINUE
111     CONTINUE
C       INSERT THE DO LOOP AND PLOTTING ROUTINE FOR   X  HERE
C       THIS REVERSES THE ORDER OF X(T)
        Z=F
        R=F/2.0
        DO 25 I=1,R
        T=X(I)
        X(I)=X(Z)
        X(Z)=T
        Z=Z-1
25      CONTINUE
        DO 10 K = 1,JK
        YW(K)=0.0
        IF(IJ .LE. G) GO TO 13
        GO TO 14
13      CONTINUE
        DO 20 I=1,IJ
        KI=(F+1.0)-I
        KY=(IJ+1.0)-I
        EUM=H(KY)*X(KI)*DT
        YW(K)=EUM+YW(K)
20      CONTINUE
        IJ=IJ+1.0
        GO TO 17
14      CONTINUE
        IF(JJ .GE. (F-G)) GO TO 16
        DO 15 JL=1,G
        KI=(F-JJ)-JL
        KY=(G+1)-JL
        EUM=H(KY)*X(KI)*DT
        YW(K)=EUM+YW(K)
15      CONTINUE
        JJ=JJ+1.0
        GO TO 17
16      CONTINUE
        II=II+1.0
        KI=(F+1)-II
        DO 21 I=1,IG
        KG=G+1.0-I
        JG=IG-I+1
        EUM=H(KG)*X(JG)*DT
        YW(K)=EUM+YW(K)
21      CONTINUE
        IG=IG-1
17      CONTINUE
10      CONTINUE
C       INSERT WRITE FOR OUTPUT YW HERE
C       INSERT DO LOOP AND PLOTS ROUTINE FOR YW HERE
C       THIS DO LOOP TAKES THE PROPER NUMBER OF POINTS FROM YW(T) FOR FFT
        DO 63 J=1,IH
        V(J)=J
        K=4*J
        AYW(J)=YW(K)
63      CONTINUE
        IF(IJV .GT. 1.0) GO TO 126
        GO TO 127
126     CONTINUE
        CALL NUFFT(IE)
        DO 64 MM=1,IH
        FYW(MM)=SQRT(AYW(MM)**2+FI(MM)**2)*DT*2
64      CONTINUE
C       REAL PART IN AYW  IMAG PART IN FI  MAG PART IN FYW
        IG=IH/2
        DO 52 J=1,IG
        V(J)=J
52      CONTINUE
        CALL PLOTS2(V,FYW,IG,1)
C       THIS FINDS THE PEAK VALUES OF FFT
        KJ=1.0
        FPKFT(1)=FYW(1)
        FPKNB(1)=1.0
        IIH=IH/2
        DO 68 J=2,IIH
```

```
      IF(FYW(J) .GT.  FYW(J-1)) GO TO 68
      IF(J .EQ. 2) GO TO 68
      IF(FYW(J-1) .GT. FYW(J-2)) GO TO 69
      GO TO 68
69    FPKFT(KJ)=FYW(J-1)
      FPKNB(KJ)=J-1
      KJ=KJ+1
      KKJ=KJ
68    CONTINUE
      WRITE(6,70)
70    FORMAT(1H1,'PEAK VALUES OF FFT')
      WRITE(6,71) (FPKNB(J),FPKFT(J),J=1,KKJ)
71    FORMAT(4(G12.7,2X,G12.7))
C     THIS FINDS THE MAIN PEAK OF FFT
      X=FPKFT(1)
      MPKFT=1
      DO 72 J=2,KKJ
      Y=FPKFT(J)
      IF(Y .GE. X) GO TO 73
      GO TO 72
73    X=Y
      MPKFT=FPKNB(J)
72    CONTINUE
C     THIS CALCULATES THE PHASE ANGLE OF THE MAIN PEAK OF THE FFT
      IIP=MPKFT
      ZIO=AYW(IIP)
      ZIT=FI(IIP)
      WRITE(6,85)
85    FORMAT(1H0,'AYW(IIP)',12X,'FI(IIP)')
      WRITE(6,86) (ZIO,ZIT)
86    FORMAT(G12.7,4X,G12.7)
      PHAGL=ATAN(FI(IIP)/AYW(IIP))*180.0/PI
      WRITE (6,74)
74    FORMAT(1H0,'MPKFT',12X,'PHAGL')
      WRITE(6,75) (MPKFT,PHAGL)
75    FORMAT(G12.7,4X,G12.7)
      GO TO 919
127   CONTINUE
      CALL NUFFT(IE)
      DO 65 J=1,IH
      Y(J)=AYW(J)
      FIY(J)=FI(J)
      FAB(J)=SQRT(AYW(J)**2+FI(J)**2)*DT*2
65    CONTINUE
C     REAL PART OF Y IN Y(J) IMAG PART IN FIY  MAG PART IN FAB
      IG=IH/2
      DO 66 J=1,IG
      V(J)=J
66    CONTINUE
      CALL PLOTS2(V,FAB,IG,1)
C     THIS FINDS THE PEAK VALUES OF FFT
      KJ=1.0
      PKFFT(1)=FAB(1)
      PKFNB(1)=1.0
      IIH=IH/2
      DO 77 J=2,IIH
      IF(FAB(J) .GT. FAB(J-1)) GO TO 77
      IF(J .EQ. 2) GO TO 77
      IF(FAB(J-1) .GT.FAB(J-2)) GO TO 78
      GO TO 77
78    PKFFT(KJ)=FAB(J-1)
      PKFNB(KJ)=J-1
      KJ=KJ+1
      KKJ=KJ
77    CONTINUE
      WRITE(6,79)
79    FORMAT(1H1,'PEAK VALUES OF FFT')
      WRITE(6,80) (PKFNB(J),PKFFT(J),J=1,KKJ)
80    FORMAT(4(G12.7,2X,G12.7))
C     THIS FINDS THE MAIN PEAK OF FFT
      X=PKFFT(1)
      MPKFT=1
      DO 81 J=2,KKJ
      Y=PKFFT(J)
      IF(Y .GE. X) GO TO 82
      GO TO 81
82    X=Y
      MPKFT=PKFNB(J)
81    CONTINUE
C     THIS FINDS THE PHASE ANGLE OF THE MAIN PEAK OF FFT
```

```
      IIP=MPKFT
      Z01=Y(IIP)
      ZTI=FIY(IIP)
      WRITE(6,87)
87    FORMAT(1H0,'Y(IIP)',12X,'FIY(IIP)')
      WRITE(6,88) (Z01,ZTI)
88    FORMAT(G12.7,4X,G12.7)
      PHAGL=ATAN(FIY(IIP)/Y(IIP))*180.0/PI
      WRITE(6,83)
83    FORMAT(1H0,'MPKFT',12X,'PHAGL')
      WRITE(6,84) (MPKFT, PHAGL)
84    FORMAT(G12.7,4X,G12.7)
      IF(IJV .GT. 1.0) GO TO 919
      PHDLY=PHAGL
919   CONTINUE
117   CONTINUE
C     THIS FINDS THE FREQUENCY
      FREQ=(MPKFT)/((G+F)*DT)
C     THIS FINDS THE PHASE SHIFT OF MAIN PEAKS
      PASDF=(PHDLY-PHAGL)
C     THIS PROTECTS AGAINST INVALID ACOS ARGUMENTS
      ARGMT=(PASDF*VLS/FREQ)/(DARY*360.0)
      BV=ABS(ARGMT)
      IF(BV .LT. 1.0) GO TO 40
      ANGLE=999.99
      ATARG=999.99
      PTGAE=999.99
      GO TO 42
40    ANGLE=(ACOS(BV))*(360.0/(2*PI))
      ATAMT=(PHSF*VLS*SPD)/(DARY*360.0)
      BB=ABS(ATAMT)
      IF(BB .LT. 1.0) GO TO 43
      ATARG=999.99
      PTAGE=999.99
      GO TO 42
43    ATARG=(ACOS(ATAMT))*(360.0/(2*PI))
      PTGAE=((ATARG-ANGLE)/ATARG)*100.0
42    CONTINUE
      WRITE(6,326)
326   FORMAT(1H0,'FFT WITH S/N = 1/INS')
      WRITE(6,327) AINS
327   FORMAT(G13.6)
      WRITE(6,323)
323   FORMAT(1H0,' FREQUENCY',5X,'AVG PHASE SHIFT',5X,'ACTUAL PHASE SHIF
     /T',5X,'ANGLE TO TARGET',5X,'ACTUAL TARGET ANGLE',5X,'% TARGET ANGL
     /E ERROR',/)
      WRITE(6,324) (FREQ,PASDF,PHSF,ANGLE,ATARG,PTGAE)
324   FORMAT(G12.7,5X,G12.7,6X,G12.7,11X,G12.7,8X,G12.7,13X,G12.7)
      STOP
      END
```

```
      SUBROUTINE NUFFT(KE)
      COMMON/HOOK/XREAL(1024),XIMAG(1024)
      NU=KE
      N=2**NU
      N2=N/2
      NU1=NU-1
      K=0
      DO 100 L=1,NU
102   DO 101 I=1,N2
      P=IBITR(K/2**NU1,NU)
      ARG=6.283185*P/FLOAT(N)
      C=COS(ARG)
      S=SIN(ARG)
      K1=K+1
      K1N2=K1+N2
      TREAL=XREAL(K1N2)*C+XIMAG(K1N2)*S
      TIMAG=XIMAG(K1N2)*C-XREAL(K1N2)*S
      XREAL(K1N2)=XREAL(K1)-TREAL
      XIMAG(K1N2)=XIMAG(K1)-TIMAG
      XREAL(K1)=XREAL(K1)+TREAL
      XIMAG(K1)=XIMAG(K1)+TIMAG
101   K=K+1
      K=K+N2
      IF(K.LT.N) GO TO 102
      K=0
      NU1=NU1-1
100   N2=N2/2
      DO 103 K=1,N
      I=IBITR(K-1,NU)+1
      IF(I.LE.K) GO TO 103
      TREAL=XREAL(K)
      TIMAG=XIMAG(K)
      XREAL(K)=XREAL(I)
      XIMAG(K)=XIMAG(I)
      XREAL(I)=TREAL
      XIMAG(I)=TIMAG
103   CONTINUE
      RETURN
      END
```

```
      FUNCTION IBITR(J,NU)
      J1=J
      IBITR=0
      DO 200 I=1,NU
      J2=J1/2
      IBITR=IBITR*2+(J1-2*J2)
200   J1=J2
      RETURN
      END
```